OASIS Project Note

NIEMOpen Publication Process Version 1.0

Project Note 01

02 May 2023

This stage:

https://docs.oasis-open.org/niemopen/niem-pubs/v1.0/pn01/niem-pubs-v1.0-pn01.md (Authoritative)
https://docs.oasis-open.org/niemopen/niem-pubs/v1.0/pn01/niem-pubs-v1.0-pn01.html
https://docs.oasis-open.org/niemopen/niem-pubs/v1.0/pn01/niem-pubs-v1.0-pn01.pdf

Previous stage:

N/A

Latest stage:

https://docs.oasis-open.org/niemopen/niem-pubs/v1.0/niem-pubs-v1.0.md (Authoritative)
https://docs.oasis-open.org/niemopen/niem-pubs/v1.0/niem-pubs-v1.0.html
https://docs.oasis-open.org/niemopen/niem-pubs/v1.0/niem-pubs-v1.0.pdf

Open Project:

NIEM Technical Architecture Committee (NTAC) of the OASIS NIEMOpen OP

Project Chair:

Katherine Escobar (katherine.b.escobar.civ@mail.mil), Joint Staff J6

NTAC Committee Chairs:

Jim Cabral (jim.cabral@infotrack.com), InfoTrack US
Scott Renner (sar@mitre.org), MITRE

Editors:

Jim Cabral (jim.cabral@infotrack.com), InfoTrack
Christina Medlin (christina.medlin@gtri.gatech.edu), GTRI

Related work:

This document is related to:

- *Conformance Targets Attribute Specification (CTAS) Version 3.0*. Edited by Tom Carlson. Latest stage: https://docs.oasis-open.org/niemopen/ctas/v3.0/ctas-v3.0.html.

Abstract:

This document describes the terminology, deliverables, versioning, stages and processes involved in publishing a NIEMOpen version, including the model, specifications and tools.

Status:

This is a Non-Standards Track Work Product. The patent provisions of the OASIS IPR Policy do not apply.

This document was last revised or approved by the Project Governing Board of the OASIS NIEMOpen OP on the above date. The level of approval is also listed above. Check the "Latest stage" location noted above for possible later revisions of this document. Any other numbered Versions and other technical work produced by the Open Project (OP) are listed at http://www.niemopen.org/.

Comments on this work can be provided by opening issues in the project repository or by sending email to the project's public comment list: niemopen-comment@lists.oasis-open-projects.org.

List information is available at https://lists.oasis-open-projects.org/g/niemopen-comment.

Citation format:

When referencing this document the following citation format should be used:

[NIEM-Publication-v1.0]

NIEMOpen Publication Process Version 1.0. Edited by Jim Cabral and Christina Medlin. 02 May 2023. OASIS Project Note 01. https://docs.oasis-open.org/niemopen/niem-pubs/v1.0/pn01/niem-pubs-v1.0-pn01.html. Latest stage: https://docs.oasis-open.org/niemopen/niem-pubs/v1.0/niem-pubs-v1.0.html.

Notices

For complete copyright information please see the full Notices section in Appendix D.

Table of Contents

1 Introduction

This document describes the means by which the NIEM reference data model and technical specifications are updated and published. It identifies processes, artifacts, and responsibilities for new versions of NIEM. It also establishes a regular cycle for predictable and manageable NIEM updates.

The NBAC manages the content of the NIEM reference data model, which is made up of Core, domains, and other namespaces. The NTAC manages the NIEM technical specifications, which define the architecture of the reference data model and the community-built extension data models and message specifications. Both TSCs work together to manage domain and community needs, maintain coherence between the model and the specifications that govern it, and to update artifacts for new NIEM versions.

1.1 Glossary

1.1.1 Definitions of terms

- **NIEMOpen** (or **NIEM**) refers to the NIEM Open Project under OASIS.
- The **NIEM community** refers to the organizations and people who use NIEM, which started in 2005.
- NIEM includes a **reference data model** (or **model**) and **technical specifications**, which govern the architecture of the NIEM model.
- **NIEM** followed by a version number (e.g., NIEM 6.0) refers to a specific version of the NIEM model and technical specifications.
- The **NTAC** is a NIEMOpen Technical Steering Committee, responsible for the technical specifications that govern the NIEM technical architecture.
- The **NBAC** is a NIEMOpen Technical Steering Committee, responsible for the NIEM business architecture, the management of the content in NIEM Core, and the support of NIEM domains. It includes representatives from NIEM domain subcommittees.
- NIEM **Core** is a namespace in the model with content that is managed collaboratively by the NBAC. This content tends to be general-purpose, widely-used, and governance does not belong to any one authoritative source.
- A NIEM **domain** refers to content in the model that is associated with and governed by a specific segment of government or industry. Its content is managed by its corresponding NBAC domain subcommittee, with oversight provided by the NBAC.
- NIEM **utility schemas** are defined by the NDR and other technical specifications to support architectural requirements in the model. They are managed by the NTAC and are also included among the schemas representing the model.
- The NBAC's **Harmonization Subcommittee** is responsible for reviewing Core and cross-domain content issues and provides recommendations for Core changes to the NBAC for approval.

1.1.2 Acronyms and abbreviations

- **CMF**: Common Model Format
- **CTAS**: Conformance Target Attribute Specification
- **HSC**: Harmonization Subcommittee
- **NBAC**: NIEM Business Architecture Committee TSC
- **NDR**: Naming and Design Rules technical specification
- **NMO**: NIEM Management Office TSC
- **NTAC**: NIEM Technical Architecture Committee TSC
- **PGB**: Project Governing Board
- **PSD**: Project Specification Draft
- **PS**: Project Specification

- **TSC**: Technical Steering Committee
- **WD**: Working Draft

Note: NIEM is no longer an acronym.

1.2 Assumptions

The NIEM Publication Process MUST comply with OP requirements (see sections 11-14) including:

- Emailing the project distribution list to announce any artifacts that require a vote of the PGB to approve (e.g. release/group releases, PSD, PS) at least 14 days in advance of the vote
- Conforming PSDs and later to the PS template, which includes methods for indicating the relevant designated branches and applicable licenses
- Ensuring any software code included in a PS is composed only of Releases or Group Releases bearing Implementer-Class Licenses.
- Ensuring any other content in a PS was contributed to a project repository

2 NIEM Versions

NIEM versions are scheduled annually on a 3-year cycle. A NIEM major version one year will be followed by minor versions the next two years. Patch versions will be published as needed. The schedule can be delayed or adjusted, however, by the PGB.

NIEM technical specifications follow semantic versioning practices. They are typically only updated for major versions.

The NIEM data model follows a variation to semantic versioning practices. This versioning strategy attempts to balance stability vs flexibility for the community by treating Core and domain changes separately:

- Core carries content that is often broadly reusable across the community. Changes to this namespace can have a greater impact on a wider range of users, so stability is more important here.

- Domains carry content to represent their own communities of interest. Domain subcommittees are responsible for the needs of their communities, are closer to their user base, and are given greater flexibility in managing their content.

As such, the versioning strategy that the NIEM model employs is that only the Core namespace follows semantic versioning:

- Major versions MAY include breaking changes to any content (including Core and domains) and the architecture (defined by the technical specifications).

- Minor versions MAY only include non-breaking changes to Core and the architecture (e.g., additions, minor documentation edits); however, **domains and other non-utility namespaces MAY include breaking changes**.

  Because older NIEM versions remain available and implementers are never required to migrate outside of their own and their data exchange partners' needs, this approach has provided a workable middle-ground to responsively meet community needs.

- Patch versions MAY only include non-breaking changes. Additions and changes to existing NIEM content are published in separate namespaces that may be used in message model schemas alongside the original data model schemas.

  Patch versions can be used for bug fixes, adding new content, and "updating" existing content. Since the breaking changes cannot be made to the original namespaces in a patch, alternate components can be published separately and can be used in message specifications as needed in place of the originals.

Note that:

- NIEM version numbers are of the form major.minor.patch, with the patch number being optional for a major (e.g., 6.0) or minor (e.g., 6.1) version.
- A NIEM version is considered ready for community use once it becomes approved by the PGB as a Project Specification.

2.1 Major Versions

Each NIEM major version:

- MUST increment the first digit in the version number.

- MAY include breaking changes to the model and technical specifications from their previous versions

- SHOULD include a new major version of the data model, with content changes managed by the NBAC and architectural changes managed by the NTAC

- MAY include major version updates to some or all of the technical specifications (e.g., Naming and Design Rules, Common Model Format), with changes managed by the NTAC

- SHOULD include a release of any normative tools or code that is required for conforming or verifying conformance with the model or specifications (e.g., NDR Schematron rules)
- SHOULD include a release of any tools or code that were instrumental to building the model or specifications
- SHOULD include a public review of the updated model and technical specification as Project Specification Drafts (PSD) before moving to further publication stages
- SHOULD progress to OASIS Project Specification(s) (PS)
- MAY be submitted for an OASIS Standard(s) in the following year and later be submitted for ISO standard(s)

2.2 Minor Versions

Each NIEM minor version:

- MUST increment the second digit in the version number.
- MAY include non-breaking changes to Core and to technical specifications from the previous versions
- **MAY include breaking changes to domains, code set namespaces, and other namespaces not imported by Core**
- MAY include updates to the normative or non-normative tools or code from previous versions
- MAY address any defects in the model or technical specifications in errata
- MAY include a public review of the updated model and any updated technical specifications as Project Specification Drafts
- SHOULD progress to OASIS Project Specification(s)

2.3 Patch Versions

Each NIEM patch version:

- MUST increment the third digit in the version number.
- MAY include only non-breaking updates to the model, with changes published in the original or in new namespaces as appropriate.
- MAY include only non-breaking updates to the technical specifications, with changes that do not affect the conformance of the model or of NIEM community message specifications.
- MAY include changes such as bug fixes, new domains or new content for existing domains or Core, and alternate representations of existing content as a means of making updates without affecting backwards compatibility.
- SHOULD progress to OASIS Project Specification.

2.4 Versioning Strategy

Specification version numbers SHOULD NOT be bumped to stay in sync with a NIEM major or minor version if the specification itself is unchanged.

Namespace version numbers in the model:

- SHOULD NOT be bumped in sync with a NIEM minor version if the namespace itself and all namespaces it references are unchanged.

  > For example, a NIEM 6.2 minor version may contain namespaces with versions remaining at 6.0 and 6.1.

- SHOULD be bumped in sync with a NIEM minor version if the namespace or any namespace it references is changed.

  > For example, the version of a namespace with new updates may be bumped directly from 6.0 to 6.2 if there were no changes to it in NIEM 6.1.

3 Project Artifacts

3.1 Artifact identifiers

- NIEM artifacts with identifiers (e.g., model schemas, specifications) SHOULD use https://docs.oasis-open.org/niemopen/ns as the root URI for resolvability.

3.2 NIEM model artifacts

The NIEM model:

- MUST have a normative representation, which for NIEM 6.0 will be XML Schemas.
- SHOULD be conformant to the reference schema document (REF) rules defined in the NDR.
- MAY include additional artifacts for documentation and developer support.

3.3 NIEM technical specification artifacts

NIEM technical specifications:

- SHOULD be edited in Markdown.
- MAY be accompanied by normative and/or informative artifacts, such as Schematron rules and examples, in the same repository or in separate repositories suitable for OASIS tool release management.
- MAY normatively define utility schemas that are included in the model repository for architecture support.

3.4 Comment resolution logs

Comment resolution logs:

- MUST be used to capture public review comments and their dispositions.
- MUST contain at a minimum the following information:
    - Link to the original comment from the NIEMOpen Feedback List
    - Name of the submitter
    - Summary of the comment
    - Disposition of the comment
- SHOULD be represented as CSV files, allowing:
    - the contents to be easily viewed on GitHub without requiring downloads
    - comparisons to be made easily to previous versions

3.5 Change comparison documents

Change comparison documents:

- MUST be maintained for any changes (material or non-material) made after the first version of a PSD was approved by the PGB before it can be submitted for approval again as the next version of the PSD.
- MUST be maintained for any non-material changes made after a PSD was approved before it can be submitted for approval as a PS.
- MAY include a link to a GitHub diff presenting a red-line comparison view of individual file changes.

4 Project Resources

4.1 Official Publication Site

Approved project work products MUST be published at https://docs.oasis-open.org/niemopen, including:

- Project Specifications and Project Specification Drafts
- Project Notes and Project Note Drafts
- Approved Errata

4.2 GitHub Repositories

NIEMOpen GitHub repositories:

- SHOULD be used as the primary means for sharing working versions of technical specification and model project artifacts with project members.
- SHOULD reside under https://github.com/niemopen.
- SHOULD provide an approved PSD as a draft GitHub release.
- SHOULD provide an approved PS as a GitHub release.
- SHOULD be leveraged for tracking issues that affect the model and specifications.

The model specification and technical specification repositories are listed below:

Specification	NIEMOpen repository
Model	https://github.com/niemopen/niem-model
Code Lists	pending
CMF	https://github.com/niemopen/common-model-format
Conformance	pending
CTAS	https://github.com/niemopen/niem-conformance-targets
NDR	https://github.com/niemopen/niem-naming-design-rules

4.2.1 Branches

- GitHub branches SHOULD be used to separate working contributions and drafts vs PGB-approved products, e.g.,
    - `main`: PGB-approved Project Specifications
    - `dev`: working drafts with incremental changes
    - Temporary or other branches MAY be created to support patch versions or otherwise as needed (ad hoc)

4.2.2 Tags

- Tags SHOULD be used to identify the NIEM version and OASIS deliverable stage, e.g. `6.0-psd01`

4.3 Other OASIS Resources

Other OASIS resources may be used to support NIEM, including:

- File repositories associated with NIEMOpen mailing lists.
- Jira, Confluence, and other OASIS-provided resources.

4.4 Legacy content and repositories

Prior to coming an OASIS Open Project, NIEM resources were managed and published at different locations.

- Legacy versions of the NIEM model MAY continue to reside at its original publication site at https://release.niem.gov.
- Legacy versions of the NIEM technical specifications MAY continue to reside at its original publication site at https://reference.niem.gov.
- Legacy GitHub repositories at https://github.com/NIEM SHOULD be archived so that the content remains available, but changes cannot be made and new issues cannot be submitted.
- Other legacy resources MAY continue to be available but changes and new issues SHOULD NOT be accepted.

5 Process Basics

The following is general information that will apply during the development and publication process of a NIEM version.

5.1 Publication Schedule

For each major or minor version, the NTAC, NBAC, and PGB SHOULD coordinate and prepare a schedule at the beginning of the year to include:

- Deadline for major content submissions from the domains and community
- Deadline to recommend approval of PSD 01 by the PGB
- Option for a public review of PSD 01
- Deadline to request a Special Majority Vote to approve PS by the PGB

The schedule MAY continue to be adjusted throughout the year as needed.

5.2 Feedback

- Community members, contributors, and maintainers SHOULD create issues in the model repository and technical specification repositories to report bugs, suggest updates, and document contributions.
- TSC repository maintainers MAY move contributed issues to a more appropriate repository.
- All issues SHOULD be addressed during the development process of a NIEM version.
    - Some issues MAY NOT be able to be resolved.
    - It is the responsibility of the TSCs to decide which issues are out of scope, will not be able to meet the schedule, may have negative effects on the community, or otherwise cannot or should not be resolved as requested.
- Public review comments from the community MUST be posted by the original reviewer to either the NIEMOpen Project Feedback List or a NIEMOpen GitHub issue; otherwise, the feedback cannot be accepted.

5.3 Updating model and technical specification artifacts

- Commits SHOULD address one issue at a time, unless the issue is tightly related with another issue.
- Pull requests MAY include multiple commits that address unrelated issues.

5.4 Data Stewards

Each domain in NIEM is governed by its own NBAC domain subcommittee, or is governed in conservatorship by the NBAC or an appointed steward. Each domain subcommittee SHOULD nominate a person or persons who are authorized to make domain content decisions and changes on their behalf.

Data stewards SHOULD also be appointed for code set namespaces and any other namespace in the model which has active representation from an authoritative source.

Before becoming approved by the NBAC, data stewards:

- SHOULD have a signed iCLA and eCLA so that contributions can be accepted.
- SHOULD have a GitHub user account so issues can be opened or responded to and pull requests can be made or assigned for review.

5.5 Public Reviews

Public reviews are optional for OASIS Open Projects before the OASIS Standard approval stage.

Requirements for a Public Review of a NIEMOpen PSD include:

- Initial public reviews MUST be held for a minimum of 15 days.
- Subsequent public reviews MAY be held for any material changes that result from an initial public review.
- Subsequent public reviews MUST be held a minimum of 7 days and are restricted in scope to only the new changes since the previous review.

The following are the steps for a NIEM PSD Public Review:

5.5.1 Decision to hold a public review

- The NBAC or NTAC SHOULD decide via lazy consensus whether to recommend a public review for an upcoming PSD.
- The NBAC or NTAC MUST record the draft title, version number, and commit link or tag in meeting minutes.
- The NBAC or NTAC MAY capture additional emails for public review notification beyond the OASIS membership.
- If the PSD and public review are approved by the PGB, the PGB MUST submit a combined request for publishing a draft and first public review to OASIS admin.

5.5.2 Public review announcement

- OASIS admin MUST announce the public review to the OASIS membership and any external stakeholders identified by the PGB.
- OASIS admin MAY also announce the public review to other public mailing lists or venues.

5.5.3 Feedback

- Non-project members MUST post any comments to the NIEMOpen Project Feedback List or to NIEMOpen GitHub issues; otherwise, the feedback cannot be not accepted.
- NIEMOpen MUST acknowledge the receipt of each comment.
- NIEMOpen MUST maintain a comment resolution log (CSV) for all public review feedback.
- NIEMOpen MUST post the disposition of each comment to its mailing list at the end of the review period.

5.5.4 Material changes resulting from a public review

If material changes result from a public review, then:

- The NTAC or NBAC MUST clearly identify changes to the draft via a change comparison document.
- The PGB MUST hold a Full Majority Vote to approve revisions as the next version of the PSD, e.g., PSD 02.
- NIEM MAY hold a subsequent 7-day minimum public review limited in scope to only the new changes by submitting a request to publish a draft with subsequent public review to OASIS admin.
- NIEM MAY continue the public review cycle until only non-material remain or all issues have been otherwise addressed.

5.5.5 Only non-material changes resulting from a public review

If only non-material changes results from a public review, then:

- NIEM MUST provide a change comparison document identifying all non-material changes.
- NIEM MAY proceed with approval of the draft as a PS.

5.6 Submitting a Working Draft for approval to the PGB

When submitting a draft to the PGB for approval as a PSD or PS:

- The NBAC and NTAC SHOULD submit drafts together when possible.
- The NBAC and NTAC SHOULD submit "Download ZIP" links from GitHub to the PGB:
  - A zip file provides an easy way for reviewers to get all necessary files.
  - GitHub can automatically handle creating zip files for specific commits and tags, which will provide a traceable record of the exact versions of files that were submitted that would not otherwise be available.

5.7 Approving a Project Specification Draft

Once issues targeted for the current version have been resolved or the schedule deadline has been reached and the artifacts have been updated:

5.7.1 Recommendation for approval

- The NBAC or NTAC, and the repository maintainers, SHOULD review the WD and make final adjustments as needed.
- The NBAC or NTAC SHOULD decide via lazy consensus when the WD is ready to recommend to the PGB as a Project Specification Draft.
- The NBAC or NTAC SHOULD decide via lazy consensus if they wish to hold a public review of the draft once approved.
- The NBAC or NTAC SHOULD notify the PGB on the PGB mailing list of their recommendations for a PSD and, if applicable, a public review. This notification should include:
  - title, version, and GitHub "Download ZIP" link
  - comment resolution log CSV if applicable (if a public review has already been held)

5.7.2 Approval process

- The PGB MUST notify the NIEMOpen community via the project mailing list at least 14 days in advance once they are ready to initiate a PGB vote or consensus call on the nominated PSD.
- The PGB SHOULD meet or hold a call for objections via a poll to approve the recommendation for PSD.
- Once approved, the PGB SHOULD submit a request to an OASIS admin to publish the PSD or, if applicable, to publish the PSD with a first public review.

5.7.3 Follow-up responsibilities

- Maintainers SHOULD add the appropriate tag (e.g., `6.0-psd01`) to the commit once the draft has been finalized and published by OASIS admin to ensure there are no non-material changes required during the publication.
- Maintainers SHOULD publish the commit on the `dev` branch as a GitHub draft release.

The PSD approval process will iterate until there are no further material changes left to be made to the draft.

5.8 Approving a Project Specification

Once the TSCs have resolved feedback on the PSD and have prepared the necessary artifacts:

5.8.1 Recommendation for approval

- The NBAC or NTAC SHOULD confirm all public review comments have been appropriately processed.
- The NBAC or NTAC SHOULD decide via lazy consensus when to nominate the designated branch to the PGB as a PS.
- The NBAC or NTAC SHOULD submit their nomination to the PGB by posting to the PGB mailing list, including
  - title, version, and GitHub download zip link
  - comment resolution log CSV if a public review was held
  - list of non-material changes made since the last approved PSD, along with an affirmation from the TSC that they judge the changes non-material

5.8.2 Approval process

- Once the PGB is ready to hold a Special Majority Vote, the PGB MUST make an announcement to the NIEMOpen Project mailing list at least 14 days in advance.
- The PGB MUST submit to OASIS Admin either:
  - a request for a Special Majority Vote to approve a Specification
  - a request for a Special Majority Vote to approve a Specification with non-material changes
- A Special Majority Vote MUST be open for at least 7 days.

Note that the date an approved vote closes is considered the specification publication date.

5.8.3 Follow-up responsibilities

- Once the specification has been published to https://docs.oasis-open.org (final non-material changes might still be required), maintainers:
  - SHOULD merge the changes in the `dev` branch in to the `main` branch.
  - SHOULD add the appropriate tag (e.g., `6.0-ps01`) to the new commit.

  - SHOULD publish the new commit as a GitHub release.

5.9 Submitting a Project Specification for approval as an OASIS Standard

- The PGB MAY decide to submit a Project Specification for approval as an OASIS Standard according to the conditions documented under OASIS Open Projects OASIS Standard Approval and External Submissions. These conditions include:
  - Three Statements of Use referencing the Project Specification.
  - A 60-day Public Review and follow-up subsequent reviews if material changes are required.
  - Submission to OASIS Members with a 14-day call for consent as an OASIS Standard.

6 Publication Processes

NIEM major and minor versions follow a schedule. Patch versions are published on an ad hoc basis.

6.1 Publishing a minor version

This section describes the stages in publishing a minor version and the steps during each stage of this process. As the technical specifications are typically not updated during minor versions, this section focuses on NBAC processes for updating the data model.

Domain subcommittees and their data stewards are responsible for leading the efforts to update their domains. The NBAC Harmonization Subcommittee and maintainers are responsible for managing updates to Core and other namespaces without data stewards, including many code set namespaces.

6.1.1 Domain content updates (breaking)

Domains MAY include breaking changes in a minor version.

6.1.1.1 Submission

Domain data stewards wishing to update their content SHOULD prepare changes in one of two ways, according to the data steward's preference:

1. Changes can be made directly to the latest draft of the domain XML schema in the model repository's `dev` branch and submitted as a pull request.
2. Changes can also be made in a specially-formatted change request spreadsheet submitted to a maintainer.

6.1.1.2 Review

A maintainer SHOULD review the submitted changes and perform quality assurance (QA) checks and other model updates, which include:

- Ensuring that the data steward has not modified content outside of the domain's control.
- Ensuring that the submitted changes conform to NDR reference schema document (REF) rules.
- Checking that the changes follow standard NIEM practices and conventions.
- Bumping version numbers of other domains or namespaces as necessary to resolve dependency issues.
- Identifying and coordinating efforts to address impacts to other domains or namespaces, other than required version number bumps.
- Looking for obvious cases of overlap with existing content.

6.1.1.3 Submission updates

A domain data steward and maintainer SHOULD collaborate on updating the initial change submission until quality assurance checks have passed.

For approved change requests submitted as updated XML schema in a pull request:

- A maintainer SHOULD accept the pull request.

For approved change requests submitted as a change request spreadsheet:

- A maintainer SHOULD ensure the domain data steward as the necessary iCLA and eCLA required for contributions to OASIS.
- A maintainer SHOULD generate updated XML schemas and submit a pull request.
- A maintainer SHOULD request a pull request review from the data steward to verify that the schema updates are correct.
- A maintainer SHOULD accept the pull request once approved by the data steward.
- A maintainer SHOULD post the final updated version of the change request spreadsheet to the file repository in the NBAC mailing list group resources.

6.1.1.4 Approval

Once changes are accepted, a maintainer SHOULD update the content in any tools needed to build reference schemas, subset schemas, and artifacts for the version.

Domain data stewards MAY repeat the process as needed within the permitted time frame for updates.

6.1.2 Other non-Core namespace content updates (breaking)

Other namespaces besides domains that do not impact Core MAY include breaking changes in a minor version. These other namespaces include code set namespaces, adapters and external standards, and auxiliary content.

Updates to these namespaces SHOULD follow a very similar process to the domain content update process described in Section 6.1.1 Domain content updates (breaking). The major difference is in the roles and responsibilities of the participants:

1. Some content will be actively managed by a participating authoritative source.
   - An approved representative SHOULD fulfill the role of the data steward and submit content updates as needed.
   - A maintainer SHOULD fulfill the same role as before, providing support and verification for the updates.
2. Some content will be managed by the NBAC.
   - A maintainer or other NBAC representative SHOULD fulfill the role of the data steward.
   - A maintainer SHOULD provide reviews of content change requests.
   - The Harmonization Subcommittee SHOULD review the changes as representatives of the NBAC before they are accepted.

6.1.3 Core content updates (non-breaking)

Core and Core-dependent namespaces (primarily utility schemas) MAY only introduce non-breaking changes in minor versions.

Non-breaking changes include:

- Creating new properties and types
- Adding additional properties (existing or new) to types (existing or new)
- Adding an existing property to a substitution group
- Adding enumerations to an existing code set type
- Expanding restrictive cardinality
- Making non-substantive definition and local terminology changes

Breaking changes, which are not permitted, include:

- Removing properties and types
- Renaming properties and types
- Removing a property from a type
- Changing the sequence of a property in a type
- Changing the type or substitution group of a property
- Changing the parent or base type of a type
- Changing or removing enumeration or other facet values on a type
- Adding non-enumeration facets to an existing type
- Removing enumerations from an existing code set type
- Making substantive definitions and local terminology changes

Maintainers and the Harmonization Subcommittee SHOULD fulfill similar roles defined in 6.1.2 Other non-Core namespace content updates (breaking) for content managed by the NBAC. Both will have the additional responsibility of ensuring that the content changes are non-breaking.

Non-breaking changes MAY be made directly in the Core namespace instead of a new additive-only namespace for Core changes. Note:

- If there are non-breaking changes to Core, the Core version number MUST be bumped to the appropriate minor version.
- If Core is bumped in a minor version, many of the other namespaces in the model will need to have their versions bumped as well due to dependency issues.
- This is a key change to the NIEM versioning strategy prior to OASIS in which any change to Core outside of major versions were automatically published to a separate Core Supplement namespace. Updating Core directly allows immediate integration of changes and requires less migration work in the future for message specification designers.

6.2 Publishing a major version

The process of publishing a major version of the data model follow many of the same processes as in 6.1 Publishing a NIEM minor versions. Additions or differences to the process include:

6.2.1 Core content changes (breaking)

The NBAC, the Harmonization Subcommittee (HSC), and the NTAC will also need to manage additional issues related to Core:

- The maintainer SHOULD prepare submitted content issues in the model repository for discussion.
- The HSC SHOULD meet regularly to address Core and cross-domain issues.
- The HSC SHOULD review Core-related pull requests.
- The HSC SHOULD make recommendations to the NBAC via the NBAC mailing list, with recommendations becoming approved after the given period of time if no concerns were raised.
- The NTAC SHOULD review architecture-related pull requests for the model.
- The NTAC SHOULD submit pull requests to make changes to utility schemas in the model based on changes to the NDR or other technical specifications.

6.2.2 NTAC specification changes (breaking)

The NTAC will manage updates to technical specifications primarily during major versions, Specification updates:

- MAY include changes to descriptive text
- MAY include changes to normative rules
- MAY include changes to Schematron or other validation languages or code to check for model and message specification conformance to the technical specification
- SHOULD include sample test cases when updating Schematron rules to support testing
- MAY require related changes to the model to remain in conformance

Major specification changes:

- SHOULD be briefed to the NBAC before implementation
- MAY be summarized as a Project Note Draft and submitted to the community for public review

6.3 Publishing a patch version

The NBAC, NTAC, or domain subcommittees MAY choose to issue patches to a major or minor version following the processes above, with the following adjustments:

- Maintainers SHOULD create a designated branch for the patch.
- Patch versions SHOULD add a third digit place to the version number, starting at 1, e.g., `6.0.1`.
- Patch versions SHOULD be submitted to the PGB for approval as a new PS.
- Patch versions SHOULD be reviewed by the original content submitters before publication but do not require public review.

Appendix A. Informative References

This appendix contains the informative references that are used in this document.

While any hyperlinks included in this appendix were valid at the time of publication, OASIS cannot guarantee their long-term validity.

- Semantic Versioning 2.0.0, https://semver.org/.

Appendix B. Acknowledgments

B.1 Special Thanks

Substantial contributions to this document from the following individuals are gratefully acknowledged:

First Name	Last Name	Company
Jim	Cabral	InfoTrack US
Tom	Carlson	GTRI
Christina	Medlin	GTRI
Scott	Renner	MITRE
Duncan	Sparrell	sFractal Consulting

B.2 Participants

The following individuals have participated in the creation of this document and are gratefully acknowledged:

NIEM Technical Architecture Committee (NTAC) TSC Members:

First Name	Last Name	Company
Aubrey	Beach	Joint Staff J6
Jim	Cabral	InfoTrack US
Tom	Carlson	GTRI
Mike	Douklias	Joint Staff J6
Katherine	Escobar	Joint Staff J6
Mike	Hulme	Unisys
Eric	Jahn	Alexandria Consulting
Dave	Kemp	NSA
Vamsikrishna	Kondannagari	DHS
Peter	Madruga	GTRI
Christina	Medlin	GTRI
Joe	Mierwa	Mission Critical Partners
Scott	Renner	MITRE
Duncan	Sparrell	sFractal Consulting
Jennifer	Stathakis	FBI
Stephen	Sullivan	BAH

Appendix C. Revision History

Revision	Date	Editor	Changes Made
niem-pubs-v1.0-pn01	2023-03-28	Jim Cabral and Christina Medlin	Initial version

Appendix D. Notices

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full Policy may be found at the OASIS website.

All contributions made to this project have been made under the OASIS Contributor License Agreement (CLA).

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

The name "OASIS" is a trademark of OASIS, the owner and developer of this specification, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, specifications, while reserving the right to enforce its marks against misleading uses. Please see https://www.oasis-open.org/policies-guidelines/trademark/ for above guidance.